AMENDMENTS TO THE CLAIMS

1. (currently amended) A method for reducing processor cycles required to send data over a communication link in packets having a packet size, the method comprising:

sending a write call, from an application running in a first context, comprising a first destination and pointing to a first quantity of data stored in virtual memory destined for ~~a~~the first destination and a second destination and pointing to a second quantity of data stored in virtual memory destined for the second destination, both the first quantity of data and the second quantity of data greater than said packet size to a driver, of a network interface controller, running in a second context through a socket;

performing a zero-copy write translating the virtual memory locations ~~to a physical memory location for~~from ~~of~~ ~~said~~ the first and second ~~quantity~~ quantities of data to ~~said the driver~~physical memory locations; and

generating, by the driver, a plurality of first packets less than or equal to ~~said~~ the packet size destined for the first destination from the first quantity of data and a plurality of second packets less than or equal to the packet size destined for the second destination from the second quantity of data, the first and second quantities of data located at the translated physical memory locations.

2. (original) A method according to claim 1, further comprising grouping data from a plurality of streams into said write call.

3. (original) A method according to claim 2, wherein said grouped data comprises all data from said plurality of streams to be sent in a time interval.

4. (original) A method according to claim 3, wherein the time interval is selected based on a bit rate of at least one stream.

5. (currently amended) A method according to claim 1, wherein ~~said zero copy write comprises writing data~~ the virtual memory comprises~~to~~ a translation buffer, the driver having a translation mapping from the virtual memory of the translation buffer to physical memory.

6. (currently amended) A method according to claim 5, wherein the zero-copy write comprises translating the virtual memory location of the translation buffer to the physical memory location using the translation mapping of the drivertranslation buffer is setup with a translation between virtual memory and physical memory.

7. (original) A method according to claim 1, further comprising generating an interrupt only after a last packet of said plurality of packets is transmitted to said communication link.

8. (original) A method according to claim 1, wherein said packet size is a maximum packet size allowable by the communication link.

9. (original) A method according to claim 1, wherein the communication link comprises a network.

10. (cancelled)

11. (original) A method according to claim 1, wherein said quantity of data comprises at least a portion of a multimedia data file.

12. (original) A method according to claim 11 wherein the multimedia data file requires real-time delivery.

13. (original) A method according to claim 11, wherein the multimedia data file is a video file, an audio file, or a game file.

14. (original) A method according to claim 1, wherein said quantity of data comprises at least a portion of a file having a format chosen from the group of formats consisting of MPEG-1, MPEG-2, MPEG-4, H.264, MP3, QuickTime, AVI, Audio/Video, real-time data in RTP format, and combinations thereof.

15. (cancelled)

16. (cancelled)

17. (currently amended) A method according to claim 1, wherein said write call comprises a write vector with entries for the first destination, the first data destined for the first destination, the second destination and the second data destined for the second destination.

18. (currently amended) A method according to claim 1, further comprising generating a single header comprising header information for a plurality of protocol layers and sending the single header to a queue for ~~a~~ the NIC.

19. (currently amended) A computer program product for sending data over a communications link in packets having a packet size, the computer program product comprising:

a computer-readable medium comprising a program module, the program module including instructions for:

receiving a write call from an application running in a first context, comprising a first destination and pointing to a first quantity of data stored in virtual memory destined for the first destination and a second destination and pointing to a second quantity of data stored in virtual memory destined for the second destination, both the first quantity of data and the second quantity of data greater than said packet size through a socket, the write call received at a driver running in a second context;

performing a zero-copy write translating the virtual memory locations for the first and second quantities of data to physical memory locations ~~of said quantity of data~~; and

generating, by the driver, a plurality of first packets less than or equal to ~~said~~ the packet size destined for the first destination from the first quantity of data and a plurality of second packets less than or equal to the packet size destined for the

second destination from the second quantity of data, the first and second quantities of data located at the translated physical memory locations.

20. (original) A computer program product according to claim 19, further comprising instructions for: grouping data from a plurality of streams into said write call.

21. (original) A computer program product according to claim 20, wherein said grouped data comprises all data from said plurality of streams to be sent in a time interval.

22. (original) A computer program product according to claim 21, wherein the time interval is selected based on a bit rate of at least one stream.

23. (currently amended) A computer program product according to claim 19, wherein ~~said zero-copy write comprises writing data~~ the virtual memory comprises~~to~~ a translation buffer, the driver having a translation mapping from the virtual memory of the translation buffer to physical memory

24. (currently amended) A computer program product according to claim 23, wherein the zero-copy write comprises translating the virtual memory location of the translation buffer to the physical memory location using the translation mapping of the driver~~translation buffer is setup with a translation between virtual memory and physical memory~~.

25. (original) A computer program product according to claim 19, further comprising generating an interrupt only after a last packet of said plurality of packets is transmitted to said communication link.

26. (original) A computer program product according to claim 19, wherein said packet size is a maximum packet size allowable by the communication link.

27. (cancelled)

28. (original) A computer program product according to claim 19, wherein said quantity of data comprises at least a portion of a multimedia data file.

29. (original) A computer program product according to claim 28 wherein the multimedia data file requires real-time delivery.

30. (original) A computer program product according to claim. 28, wherein the multimedia data file is a video file, an audio file, or a game file.

31. (original) A computer program product according to claim 19, wherein said quantity of data comprises at least a portion of a file having a format chosen from the group of formats consisting of MPEG-1, MPEG-2, MPEG-4, H.264, MP3, QuickTime, AVI, Audio/Video, real-time data in RTP format, and combinations thereof.

32. (cancelled)

33. (cancelled)

34. (currently amended) A computer program product according to claim 19, wherein said write call comprises a write vector with entries for the first destination, the first data destined for the first destination, the second destination and the second data destined for the second destination.

35. (currently amended) A method for reducing buffering requirements on a network switch; the method comprising:

receiving a write call from an application running in a first context through a socket, the write call comprising a plurality of destinations, including a first destination and a second destination, and pointing to a first quantity of data destined for the first

destination, and pointing to a second quantity of data destined for the second destination, the write call received at a driver running a second context;

packetizing said first quantity of data into a plurality of packets less than or equal to a packet size, each packet destined for the first destination;

generating at least one packet comprising at least a portion of the second quantity of data destined for the second destination;

transmitting a first packet destined for the first destination to the network switch; and

transmitting the at least one packet destined for the second destination to the network switch, before transmitting a second packet destined for the first destination.

36. (original) A method according to claim 35, wherein the first quantity of data comprises at least a portion of a video media file.

37. (original) A method according to claim 35,
further comprising generating a plurality of said packets comprising at least a portion of the second quantity of data destined for the second destination.

38. (original) A method according to claim 35, further comprising:
transmitting the second packet destined for the first destination.

39. (cancelled)

40. (original) A method according to claim 35, further comprising: communicating at least one of said packets to a network interface card.

41. (currently amended) A method according to claim 35, wherein the write call points to the first and second quantity of data stored in virtual memory and further comprising: performing a zero-copy write translating the virtual memory locations for the first and second quantities of data to physical memory locations ~~of said quantity of data~~.

42. (original) A method according to claim 35, wherein said write call comprises a write vector.

43. (currently amended) A computer program product for balancing load on a network switch, the computer program product comprising:

a computer-readable medium comprising a program module, the program module including instructions for:

receiving a write call from an application running in a first context through a socket, the write call comprising a plurality of destinations, including a first destination and a second destination, and pointing to a first quantity of data destined for the first destination, and pointing a second quantity of data destined for the second destination, the write call received at a driver running in a second context;

packetizing said first quantity of data into a plurality of packets less than or equal to a packet size, each packet destined for the first destination;

generating at least one packet comprising at least a portion of the second quantity of data destined for the second destination;

transmitting a first packet destined for the first destination to the network switch; and

transmitting the at least one packet destined for the second destination to the network switch, before transmitting a second packet destined for the first destination.

44. (currently amended) A system for sending data across a network, the system comprising:

a computer running an application in a first context configured to send a write call to a driver running in a second context through a socket, the write call comprising a first destination and pointing to a first quantity of data stored in virtual memory destined for the first ~~a~~ destination and a second destination and pointing to a second quantity of data stored in virtual memory destined for the second destination, both the first quantity of data and the second quantity of data greater than said packet size to a driver, of a network interface controller, running in a second context, the application further configured to perform a zero-copy write to translate the virtual memory locations for the first and second quantities of data to physical memory locations~~of said quantity of data to said driver~~;

the driver configured to generate a plurality of first packets less than or equal to the packet size destined for the first destination from the first quantity of data and a plurality of second packets less than or equal to the packet size destined for the second destination from the second quantity of data, the first and second quantities of data located at the translated physical memory locations~~packetize said data into a plurality of packets less than or equal to said packet size~~; and

a downstream device adapted to receive at least one of said packets.

45. (original) A system according to claim 44, further comprising a network switch coupled to the computer and the downstream device, adapted to receive at least one of said packets and route the received packet to the downstream device.

46. (original) A system according to claim 44, wherein the downstream device comprises a down stream device has a timing requirement for receipt of data.

47. (original) A system according to claim 45, further comprising a plurality of said computers in communication with the network switch.

48. (original) A system according to claim 44, further comprising a plurality of downstream devices in communication with the network switch.